2012

The Information Paradox

Denning, Peter J.

The Information Paradox. 2012. (With Tim Bell) American Scientist, Nov-Dec. Classical information theory has no room for meaning -- but humans persist in assigning meaning. How can we reconcile this difference?

http://hdl.handle.net/10945/35461

The Information Paradox

Classical information theory has no room for meaning—but humans persist in assigning meaning. How can we reconcile this difference?

Peter J. Denning and Tim Bell

Since mathematician and communications engineer Claude Shannon developed information theory in the 1940s, the study of information has advanced rapidly. But along with the development of the field and its widespread effects on people's lives has appeared a series of perplexing questions. Most scientific disciplines rely on information-processing methods to discover new knowledge, and many scientists now say that information processes appear in nature. Even so, our models of these processes assume that the processes evolve without depending in any way on the meaning of the information they contain. How can such processes generate new knowledge?

Shannon's classical information theory demonstrates that information can be transmitted and received accurately by processes that do not depend on the information's meaning. Computers extend the theory, not only transmitting but transforming information without reference to meaning. How can machines that work independently of meaning generate meaning for observers? Where does the new information come from?

Software designers, scientists and consumers look to software to generate outputs that mean something—memorable experiences, new discoveries, pension projections, love letters, inspiring images and much more. But the meaning of the information seems to depend on the observer. For example, a tabulation of stock prices may look like numerical gibberish to a financial amateur but a source of riches to a professional investor. How can there be a science of information if its fundamental objects are at least partly subjective?

These questions look paradoxical because information theory says that meaning is irrelevant—a statement that our experience with computational systems seems to contradict. Moreover, the concept of information seems fuzzy and abstract to many people, making it hard for them to understand how information systems really work. Our objective in this essay is to show that information is real, existing as physically observable patterns. We review what information theory has to say, how it combines with computability theory and where its limits are. This examination shows why classical information theory cannot explain phenomena such as meaning and generation of new information. We describe a model that resolves the paradox.

Communication Systems

The simplest kind of information system is a *communication system.* In a 1948 paper titled "A Mathematical Theory of Communication," Shannon offered the first theoretical model of such a system *(see Figure 2).* At its essence is the following process: A source sends a message. An encoder generates a distinct signal for the message, as prescribed in a codebook. The channel is the medium that carries signals from the source to the receiver. A decoder on the receiver end converts the signals back to their original form, using the same codebook—and the message has arrived. Shannon's model applies to any system that encodes, decodes, transports, stores or retrieves signals or data. It also serves as a model of scientific discovery, treating it as the communication of previously unknown facts.

An important element of the communication model is *noise*—any disturbance in the channel that alters a signal, causing the decoder to output the wrong message. Examples of noise abound in communication technologies: Fog and darkness interrupted ship-to-ship semaphore communications; excessive distance between telegraph operators degraded the signal strength; lightning strikes disrupt AM radio transmissions; scratches on a CD render it unreadable; and environmental sounds drown out speech.

Note that in communication systems, coding is not the same as encryption. Encryption is an additional step that converts messages from the source into cipher text before that text goes to the encoder so that only those who have the cipher keys can read them. The job of the communication system is to deliver the cipher text accurately to the receiver, which can then remove the cipher if it has the key.

Shannon introduced the term *bit* (short for binary digit) to describe the basic unit of information. Information can be represented as patterns of bits, a process called *digitization*—literally, the conversion of analog information into digits. Digitization does not result in an exact replica of the information; it is an approximation that frequently loses some of the information. Some examples are obvious, such as pixelated photographs in which every object has ragged edges; others are more subtle. Quantities from physical phenomena, such as the orbital position of a Mars lander, cannot be represented exactly in the finite arithmetic of the computer. Rounding errors can accumulate over many computational steps, placing the accuracy of the overall computation in doubt. Even worse, some com-

Peter J. Denning is director of the Cebrowski Institute for Innovation and Information Superiority at the Naval Postgraduate School in Monterey, California, and is a past president of ACM (the Association for Computing Machinery). Tim Bell is a professor and deputy head of department in Computer Science and Software Engineering at the University of Canterbury, Christchurch, New Zealand. He directs the CS Unplugged project (http://csunplugged.org). Denning's e-mail: pjd at nps dot edu

Figure 1. In 1956 IBM introduced the world's first magnetic disk information-storage system, the RAMAC 350 *(shown above).* A promotional film for the machine showed hallways of file cabinets with frazzled secretaries walking along them. The RAMAC 350, the film showed, stored the entire content of the cabinets in about 2 cubic yards and allowed near-instantaneous searches of the data. To contemporary viewers, the film also demonstrates that the struggle to make the seemingly abstract concept of information tangible is not new. (Photograph courtesy of IBM.)

putational steps can magnify errors; for example, the difference between two almost-equal numbers can round to zero and then cause a major error when divided into another number. Designers of mathematical software have devised many tricks to prevent digitization errors from wrecking their results.

But as Shannon pointed out, many communication systems need not suffer from digitization errors. Every continuous, bandwidth-limited signal can be digitized without any loss of information by sampling at twice the highest frequency. Audio compact discs (CDs), for example, record 44,000 samples per second without significant loss of quality, because the human ear cannot hear sounds whose frequency is greater than 22,000 hertz.

Information is real and observable in all communication systems. Bits are abstractions that we use to specify what we want the systems to do. But all the components are physical, and information is always encoded into some sort of signal, which can be transmitted and translated without losing the information it encodes *(see Figure 3).*

Because information is always represented by physical means, it takes time and energy to read, write and transform it. Communication and computation can never be free of the constraints of the physical world. Computer chip engineers know that effects such as heat accumulation and feature size (the average size of the various elements contained on a chip) place real limits on how small they can make their circuits. And the time cost for every operation places limits on how many instructions can be computed in the time available. Although new algorithms have yielded dramatic improvements in finding optimal results for common problems, larger cases are intractable. For example, finding the two prime factors that make up a 600-digit key for the widely used RSA encryption system would take centuries on the fastest known computers.

Our ability to store and compute information has increased exponentially over the years. In the same year Shannon published his essay, and in the same place—Bell Labs—the newly invented transistor began to replace vacuum tubes in electronic computers. Circuit designers were able to compress the size of transistors, putting about twice as many into the same physical space at no additional cost every 18 months. They have been doing this year after year for nearly 50 years, giving us 100 times more computational power with every decade. This trend is known as Moore's Law after Intel cofounder Gordon Moore who first described it in a 1965 paper.

Moore's Law has given us two effects. One is amazing computational abilities that would appear as magic to the 1940s pioneers of computing science. The other is a flood of information, as James Gleick calls it in his 2011 book *The Information: A History, a Theory, a Flood*. The first effect is concerned with the accumulation and transmission of information and the second with meaning in our lives.

Those vast computational abilities have given rise to the popular notion that, because computation manipulates ethereal bits, not atoms, there is no physical limit on the size and power of computational structures. This notion is dead wrong. An abstract bit can do nothing until it is recorded in a physical medium, where a machine can get at it. The recording process brings us back to the world of atoms: We cannot have computation without them. We can make computation, transmission and storage breathtakingly small and fast, but we will never completely eliminate their time and energy costs.

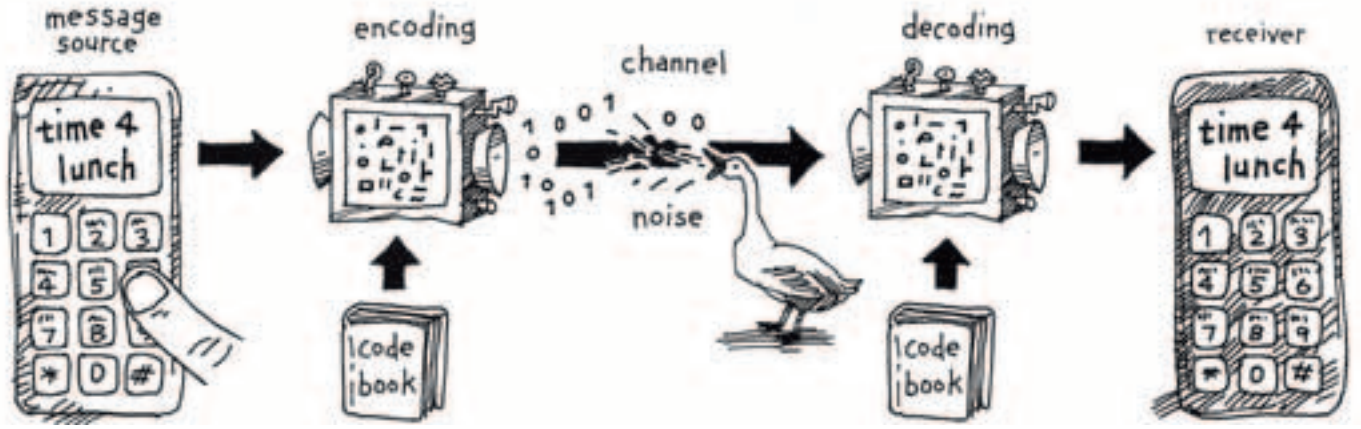


Figure 2. Claude Shannon (1916–2001) described a model information system that is now the basis of information theory. The message source represents the set of all messages that could be sent. The channel is the physical medium for storing and carrying signals. Encoding converts messages into signals, and decoding converts signals back into messages. The codebook is the rules for converting messages to signals and back again, and noise is any disruption that alters the signals.

Measuring Information

Shannon wanted a method to measure the information inherent in a source. The number of bits in a code cannot serve as such a measurement, since a single source can be represented by multiple codes. He wanted to know the shortest possible code for a set of messages. A code with fewer bits would fail to transmit some messages.

He rejected measures that depended on the meanings human observers assign to signals, and instead sought mechanisms for encoding, transmitting and decoding that worked the same every time, regardless of the context in which they were used. Postal services follow a similar principle: Their distribution and delivery systems do not depend in any way on the contents of the envelopes they transport. Shannon's remarkable insight was this: He equated the reception of information with the reduction of uncertainty. He defined information as the minimum number of yes–no questions needed to determine which of many possible messages a source was sending. The more we know about what a source might send, the less information we gain when we see what it sends.

Imagine that you know someone will respond only Y (yes) or N (no) to a question, but you have no way of knowing in advance which answer the speaker will give. You have uncertainty between the two choices Y and N; the speaker resolves your uncertainty by saying Y or N. Shannon would say that the speaker just gave you one bit of information (either 0 or 1), which selects the actual response from the two possible responses. When there are more than two choices, more bits will be needed to distinguish the message that was sent.

Suppose that we want to find the page containing a friend's name in a phone book. How many bits do we need to encode the page number? A clever method answers this question. We open the book in the middle and ask which half holds the name, which is easy to do since the contents are alphabetized. We then split that half in half using the same question. We repeat this step until only one page remains. The friend's name should appear on that page. The repeated question ("Which half?") takes us rapidly to the location. With a 512-page phone book, the first question leaves us with 256 pages to search, the second question leaves 128 pages, then 64, 32, 16, 8, 4, 2 and, finally, 1. It takes 9 yes–no questions to find the page containing the word. Therefore, when we learn the page number that contains our friend's name, we have received 9 bits of information.

In constructing a code, people take account of the probabilities of occur-

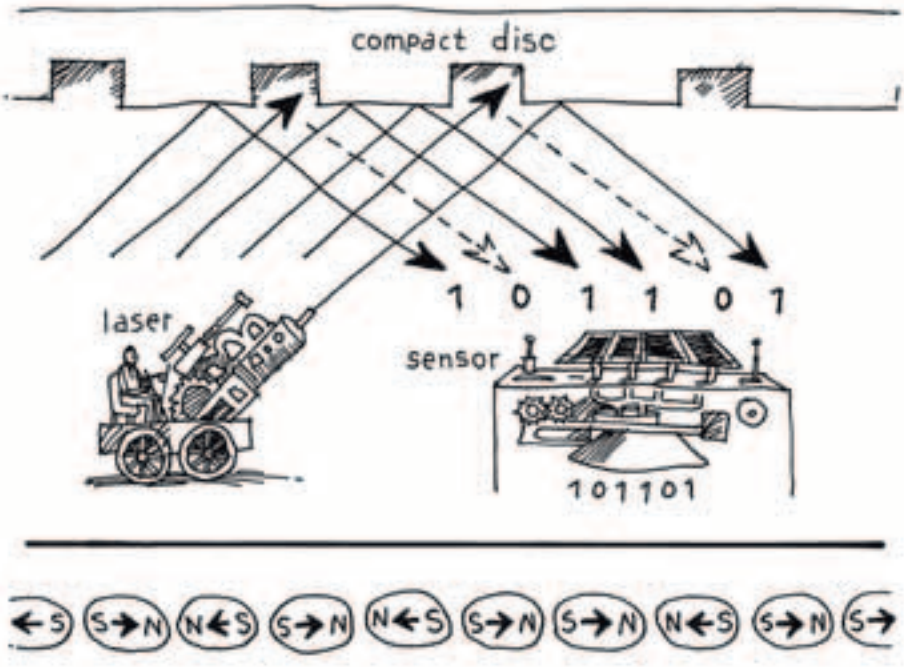


Figure 3. Information exists as patterns in physical media. "0" and "1" are labels for two distinct states of the medium. As a laser beam passes along the surface of a compact disc *(top)*, the reflection from areas called *lands* is converted by a sensor into an electrical signal for 1. Impressions in the surface, which are less reflective, are called *pits* and are sensed as 0. On a computer's hard disk *(bottom)*, binary data are stored using the direction of magnetization of particles on the surface.

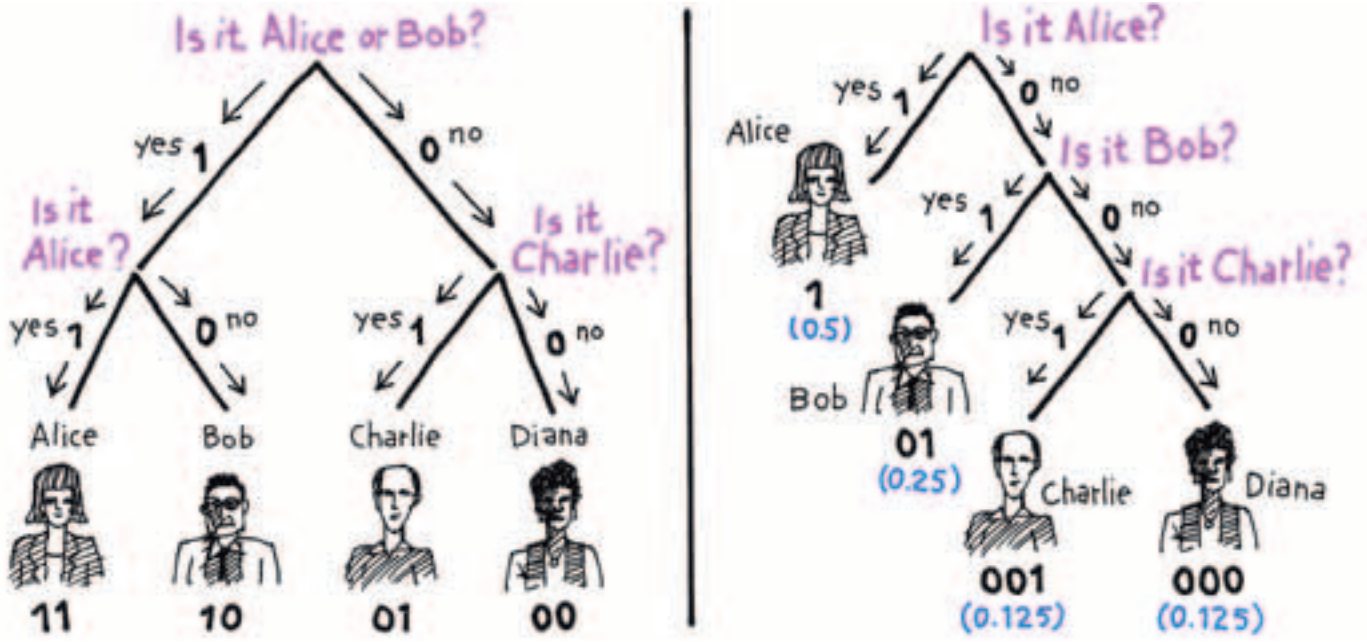


Figure 4. Shannon defined the amount of information contained in a message as the number of yes–no questions needed to select the message from the source. The questions reduce uncertainty about which message is sent. It's a convention of the field to refer to the variables that represent messages as "Alice," "Bob," and so on through the alphabet. Imagine, for instance, that we need to find which person has been selected to do a task. Using a simple decision tree *(left),* we ask, Is it Alice or Bob? If the answer is yes, the selection will be in the left half of the tree. One additional question, Is it Alice?, reveals the answer. The code for each individual is the path describing the yes-no patterns that lead to him or her. If we know the probability that an individual will be selected *(at right, blue text),* we can make a graduated decision tree that results in codes of variable length. For instance, if Alice is most likely to be chosen, we assign a code of 1. Bob, the next most likely, gets 01, and Charlie and Diana, who have equal probability, both get 3-bit codes.

rence of possible messages. Samuel Morse, who devised Morse code to use with the electric telegraph he coinvented in the 1830s, assigned the shortest code—a single dot—to the letter *e* because he knew that *e* is the most common letter in English (about 12 percent of all letters used). He assigned the longest code to the letter *j* because it is one of the least common letters (about 0.15 percent). These choices minimized the average length of a transmission. Figure 4 shows how the questions one must ask to identify a message can define a code for the message, and how prior knowledge of the probability of occurrence of various messages can lead to shorter codes.

Suppose that we have a set of code words of lengths *Li* and probabilities *Pi.* The average length *(L)* of the code is

$$L = \sum \{Li * Pi\}.$$

For the code in Figure 4, this formula gives 2 bits for the average length of the first code and 1.75 bits for the second.

What are the lengths of the codewords in the optimal code, that is, the one that minimizes *L?* Shannon answered that question in an appendix to his 1948 paper, in which he showed that the optimal length of a code word is –log *Pi,* the negative base-2 log of *Pi*'s probability. Therefore, the average length of the optimal code is

$$L = -\sum \{Pi * \log Pi\}.$$

This formula has the same form as the entropy formula of thermodynamics—and a similar interpretation. Entropy is a measure of disorder or uncertainty about the state of a system. The more disordered a set of states is, the higher the entropy. The greatest disorder occurs when all states are equally likely to occur. The greatest order occurs when one state is certain and the others do not occur at all.

Shannon considered entropy to be the measure of the information in a source. A source consists of a set of possible messages and their probabilities of occurrence. The entropy, which depends only on the probabilities of the messages, not on their codes, tells us the average length of the shortest possible code. Any shorter code would be ambiguous and could not be uniquely decoded. Take the following example:

A: 1 B: 0 C: 01 D: 10

If these messages have probabilities of 0.5, 0.25, 0.125 and 0.125, respectively, the resulting code will have an average length of 1.25 bits. However, a receiver would not be able to tell whether 1001 stands for ABBA, ABC, DBA, or DC. The entropy of the messages (calculated using the formula above) is 1.75, which defines the threshold between decipherable and indecipherable codes.

Another way to put this is that the entropy threshold defines the boundary between reliable and unreliable channels. If the source sends a new message every *T* seconds, and the shortest code has average length *L,* the source is generating a demand of *L/T* bits per second. If the channel bandwidth is *L/T* or higher, then all the bits offered by the sender can flow to the receiver. If the channel bandwidth is lower than *L/T*, some bits will be lost and the receiver will be unable to recover the original messages.

In 1951, David Huffman at MIT devised an algorithm for generating a code of minimum average length given the message probabilities. His method generates a code whose average length is within one bit of the entropy threshold. In the examples shown in Figure 4, Huffman's algorithm generates the first code when all the messages are equally likely and the second code when they have the unequal probabilities given in the example.

Another important application of information theory is file compres-

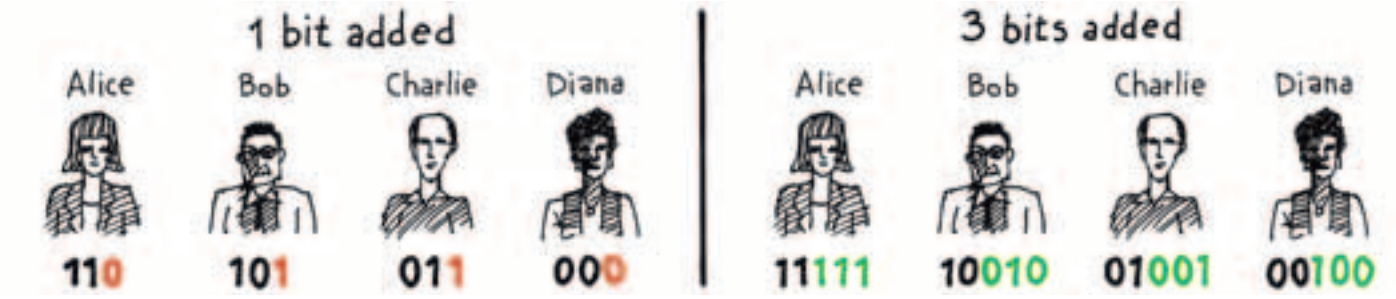


Figure 5. Short codes are easily disrupted by noise. A single-bit error in a two-bit code can change one code word into another. To alert the receiver to errors, we use parity bits. At left, a parity bit *(red)* gives each code word an even number of 1s. Now a one-bit error will result in three bits with an odd number of 1s, a noncode pattern that the receiver can detect. With three parity bits *(at right, green)*, a one-bit error causes the code word to differ by one bit from the correct code word and by two or more bits from the other code words. The decoder can thus detect and correct corrupted code.

sion, which reduces storage space and transmission time. Most computers represent text with fixed-length codes. These files can often be shortened by half by finding repeating patterns and replacing them with shorter codes. The file formats ".zip" and ".rar" employ this strategy. These formats use lossless compression: The original files can be completely recovered.

Another category of methods is called lossy compression. These approaches offer much greater compression factors but cannot entirely recover the original file. For example, MP3 audio compression reduces file size by a factor of 10 after discarding frequencies that most people are unlikely to hear. JPEG image compression discards bits that generate colors barely discernible by the human eye. Such compression schemes enable the economical sale to consumers of DVDs, online movies and music recordings. The small loss in quality incurred by these methods is usually considered a good trade-off for the large reduction in file size.

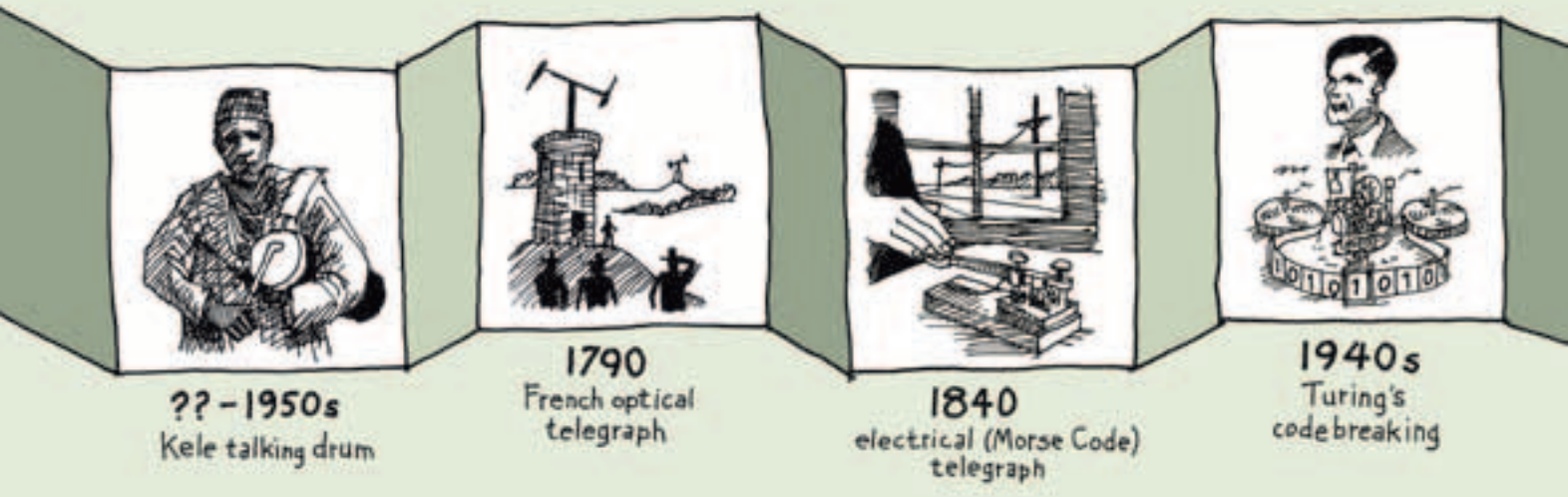
Transforming Information

A pure communication system simply transmits information from one place to another. But most computers do more than simply transmit: they *transform* information. Transformation opens many new possibilities, most notably the creation of new information.

Simple transformations include squaring a number, calculating a specified number of decimal digits of pi, and arranging a list of numbers in ascending order. Each takes a pattern of information as input and creates a pattern of information as output. In mathematical terms, a transformation is a *function.* Any function that can be programmed on a computer is called a computable function. One requirement for computability is that the function finishes its transformation in a finite time—that is, without getting stuck in an infinite loop.

Computers combine computable functions with communication channels. A channel brings the input pattern to the machinery that computes the function; another channel brings the output pattern to its destination. In these scenarios, nothing is added to the information; it has only been transformed. Yet a function can have a very simple input and generate a large number of digits as its output. For example, the pi-computing function above would generate 900 digits of pi as output in response to an input of "900." Information theory says that these digits are not extra information, but to many observ-

We may think of communication systems as a product of modern technology, but as James Gleick relates in *The Information* (2011), they prefigure the industrial era. Gleick cites the Kele people, of what is now the Democratic Republic of the Congo, whose talking drums mimicked speech and were used for long-distance communication. Of course, the Industrial Revolution sped developments. The Napoleonic government used lights on tall towers to send encoded messages. Samuel F. B. Morse developed an electrical telegraph—and the code to go with it—to send signals greater distances. Shortly thereafter, Gugliemo Marconi (and others) pioneered radio transmission of electrical signals. By the 1940s radio and telephone engi-



ers, the digits are new, useful information. Other transformations, such as averaging or sorting, can also provide new information to an observer, even if the number of output bits is smaller than the number of input bits.

A computer is a machine controlled by a program of instructions. Both the program and its data are patterns of bits—that is, information. In other words, the computer is a machine that uses some information to control how it transforms other information. This simple fact imposes two very strong constraints on what a computer can do. First, as Alan Turing showed in 1936, there are functions that no computer can evaluate. Turing's example was the halting problem—no program exists that can inspect the code of any given program and tell whether it contains an infinite loop. A modern example is malware detection—there is no program that can tell whether any given program has a malicious procedure embedded within it. The existence of noncomputable functions is a limitation imposed by information itself. Second, since information always has a physical representation, every computational step in a program requires time and energy. Many computable functions require so many steps that there is not enough time for them to return answers within any deadline we can live with. They are computable but intractable—for instance, the factoring of an RSA encryption key.

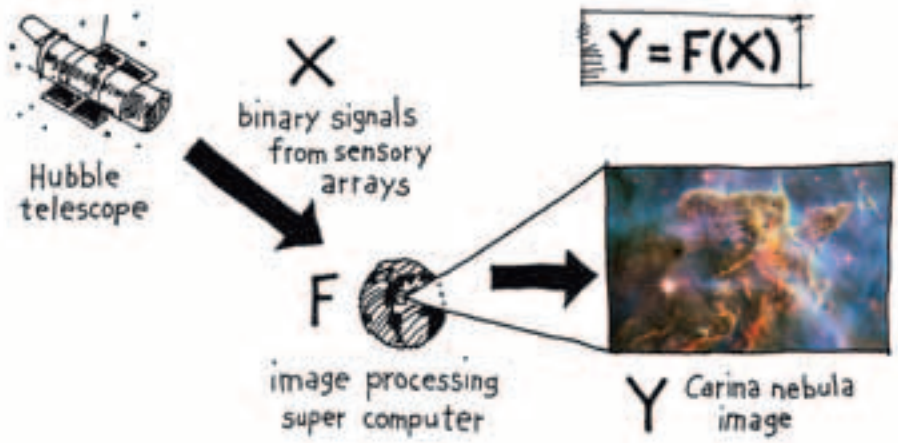


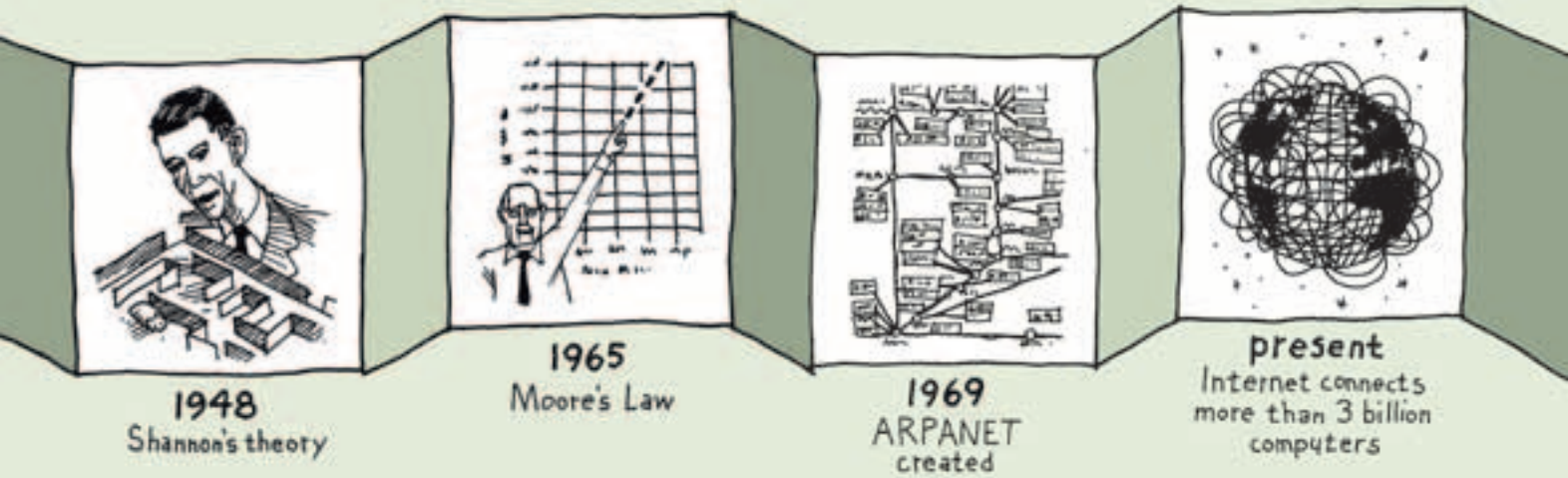
Figure 6. The Hubble space telescope's photon-gathering sensor arrays encode terabytes of data for transmission to Earth. The data are then processed into images. Computing theory would characterize the image processing as a function *f* applied to the input data *x*. The machine implementing the function, and the signals sent to it and generated by it, do not depend in any way on the meaning of the information from the telescope. Yet human observers see *y*, the output, as a beautiful image—of the Carina nebula, in the instance above. (Image courtesy of NASA.)

Even when we restrict our attention to computable functions that return answers soon enough to be useful, we find interesting questions. When a function computes bits we have not seen before, are those bits new information? Or are they just the consequence of existing information? Does DNA contain information? Many biologists say it does. If it's a message, who is it from and to? If we decode DNA, do we gain information? The decoded DNA might be used to find a cure for a genetic disorder, or it might identify the perpetrator at a crime scene. Does matching the DNA to a database merely uncover existing information, or does it generate new information? Questions like these cannot be answered with classical information theory.

Interaction Systems

Many computers programs are not simple functions: They can receive new input and generate new output at many points, and they may go on doing this without end. These *interaction systems*, as they are called, are everywhere. Every operating system is an interaction system, as are a car's GPS

neers had introduced digital sampling methods. During World War II, Alan Turing famously used an early computer to help the British crack the Germans' Enigma code. Claude Shannon's framework for coding and decoding provided a theory of information. In 1965 Gordon Moore formulated Moore's Law, which predicted the explosive growth of computing power over the next decades. The creation by the U.S. Defense Department of ARPANET marked the beginnings of the Internet. Four computers were connected in 1970; by 1981 the network had 213 hosts. Today the Internet connects more than 3 billion computers and by 2016 is expected to carry more than 1.3 zettabytes (more than 1 trillion gigabytes) of data each year.



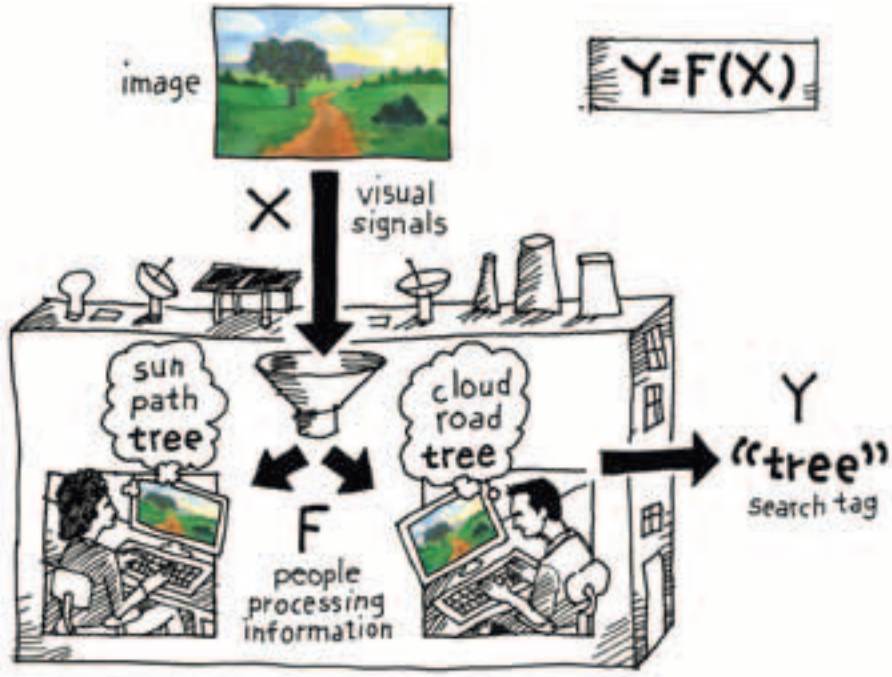


Figure 7. In a 2004 paper, Luis von Ahn and Laura Dabbish of Carnegie Mellon University described a novel computer game. In the ESP Game, players are paired and shown an image. They type words that describe the image with the goal of finding a word their (unknown and unseen) partner has also used. The common word becomes a new search tag for the image. The game teams up humans with machines to compute a function (image recognition) that no one knows how to compute by machine alone. Like other functions, it transforms information, but now the meaning interactively supplied by the players shapes portions of the transformation.

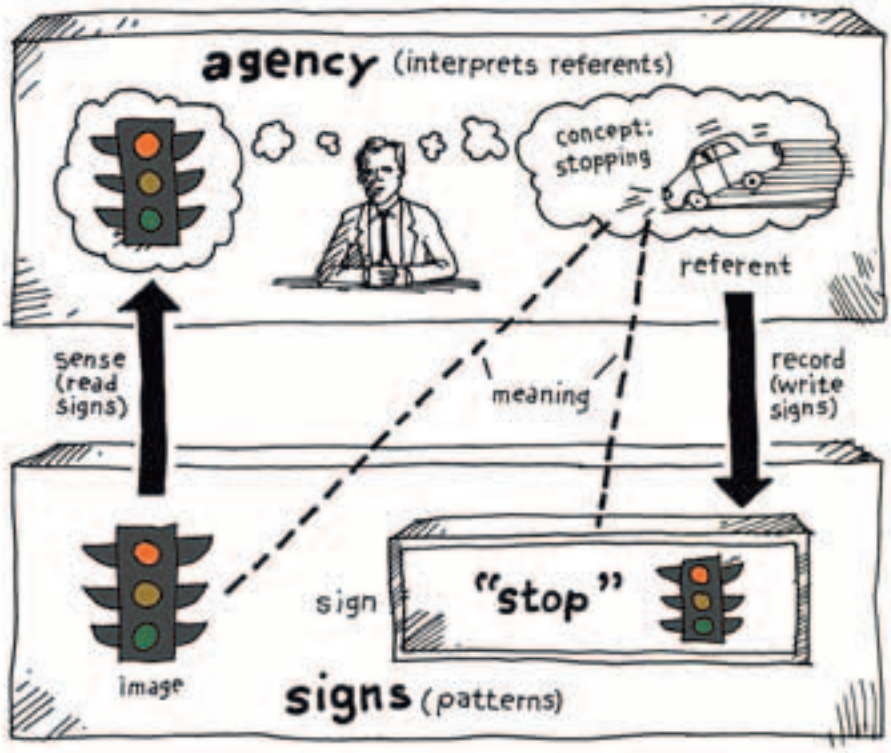


Figure 8. An alternate definition of information accommodates meaning: Information consists of a pattern, or *sign (S),* and a concept or thing, the *referent (R),* associated together. The association between *S* and *R* is the meaning. The agency is an organism (or higher-level machine) apart from the machine that stores the sign. Agency supplies the meaning that the sign-containing machine cannot. This definition provides for the coexistence of meaningless signs and meaningful referents.

system, Facebook, or an online merchant's Web server. The Internet is a global interaction system for exchanging data and coordinating actions; the Internet's domain name system (DNS) is also such a system. A distinguishing feature of interaction systems is that they operate continuously; they have no finite end. In contrast, function systems are finished when they produce their answers. In his 1986 book, *Finite and Infinite Games,* philosopher James Carse applied the metaphor of games to many realms of human affairs, writing, "A finite game is played for the purpose of winning, an infinite game for the purpose of continuing the play." Function machines are finite games; interaction systems are infinite games.

For years there was intense debate among computer scientists about whether interaction is more powerful than function computation. In recent years, experts have come to agree that interactive computation is more powerful. The contemporary conundrum of how to label digitized images illustrates why. The solution to this problem lies in interaction: A game structures the interaction between humans and machines to perform a function that no human or machine could do alone.

Interaction systems can generate outputs that no known computable function can. How do such systems create new information where none seemed to exist before?

Information and Relativity

Computing without reference to meaning works for communication channels but not for computation in general. People do not pay to play the online game World of Warcraft because they know that the signals encoded by Blizzard Entertainment will be accurately received on their screens; they do so because they want the experience of mastering quests and joining with friends to defeat evil dragons. What they are paying for is meaning. A software designer designs meanings for the users of the software; the same is true in science. The CERN team would not have built the Large Hadron Collider had they believed that the output of its computers would be meaningless.

In the early 1900s, Albert Einstein struggled with an apparent contradiction. The laws of physics said that the speed of light would depend on the motion of the observer relative to the light source. The Michelson-Morley experi-

ment said that light speed is independent of the motion of the observer. Einstein's solution, which became known as relativity theory, postulated that both apparently contradictory aspects are simultaneously true: Motion can be measured only relative to the observer, and light has a constant speed everywhere. Can we apply the "simultaneously true" idea to our problem of reconciling the apparently contradictory notions of objective and subjective aspects of information?

In a 2010 study of a large variety of signals and machines that process them, Paolo Rocchi found a model that simultaneously accommodates the subjective and objective sides of information: Information always has two parts, *sign* and *referent*. Meaning is the association between the two. This provides the basis for a reconciliation.

An association between a sign and its referent can be stored in our brains; when we see the sign, we know what it means. For example, when we see the image of a red light (sign), our brains command "stop" (referent). An association can also be written down as a new pattern: *red light*, *stop*. Once recorded, the association can be transformed or processed by a machine. The pattern *red light*, *stop* might be used by a robot-driven car.

Sometimes it is not obvious how to describe a referent. Years ago, it was thought that only our brains could recognize faces; the task was impossible for computers. Then we learned to describe faces as combinations of features that could be sensed in a digital image, and to describe an association between a combination of measurable face features and the name of a person. Computers can now perform automatic face recognition.

The progression of science illustrates the development of new information. When we come to understand a phenomenon, we are able to describe the association between its signs and its effects, and we can use a machine to find instances of that association.

Because Shannon ignored meaning, his theory could not explain where new information comes from. Rocchi's model states that a new association between a sign and referent is new information. Tim Berners-Lee, in his 2000 book *Weaving the Web,* makes a similar suggestion about the Internet: Someone who creates a new hyperlink creates new information and new meaning.

The sign-referent model also explains how we know what it means when a computer program generates results we have not seen before. Suppose, for example, that we are given a set of input-output pairs *(x,y)* observed in past performances of an experiment. Using statistical regression, we can find the best parameters *a* and *b* for a straight line fitting the data: $y = ax + b$. We can then use the straight line to predict the output *y* that will be generated for a new value *x*. The program's output is a formal description of a straight line. The meaning is that the line is a trend in the data that can be used to make predictions.

Bayesian inference is a much more sophisticated method of analyzing data. It says that the probability of a hypothesis *H* given evidence *E* can be computed from the probability of *E* given *H*, the probability of *H,* and the probability of *E*. A diagnostician wanting to know the probability that a patient has encephalitis, given the symptom of neck pain, can compute it from the known fraction of encephalitis patients with neck pain, the fraction of patients with neck pain and the fraction of patients with encephalitis. Bayesian inference programs are extensively used in data mining—they can infer complex hypotheses using the evidence in very large data sets.

In classical information theory, we would say that Bayesian inference works by determining the content of a message source based on data observed from the source. For communication of messages, it was reasonable for Shannon to assume that the content of the source was known a priori. For scientific discovery, the set of possibilities within the source is initially unknown; the discovery makes them, and their probabilities, known. Bayesian inference is an automated method of transforming observed data from a source into knowledge of the content of the source—of creating new information.

Resolving the Paradox

Human beings have been encoding signals for transmission in different media channels since time immemorial. In the 1940s, Shannon's theory of information allowed communication and computer engineers to design digital communication systems so that no information would be lost and errors introduced by noise could be removed.

As our ability to store information multiplied exponentially, as Moore's Law had predicted it would, we were increasingly pressed to say what all that information means. By definition, Shannon's information theory could not resolve the question. Is a never-seen-before result new information or simply a new pattern made from existing information? There is the paradox: How can a system process information without regard to its meaning and simultaneously generate meaning in the experience of its users?

The idea that information consists of both signs and referents resolves this paradox. The association between a sign and its referent is new information. Through the links we establish on the Web and in the outputs of all the programs we design, we constantly create new information, in vast quantities—and people assign meanings to it. The sign-referent interpretation reconciles this reality with information theory. It lets us celebrate the role of designers, who anticipate the meanings of computations.

Bibliography

Bell, T., and M. Fellows. CSunplugged.org website. Video presentation of a workshop. http://csunplugged.org/videos.

Berners-Lee, T. 2000. *Weaving the Web.* Harper Collins.

Carse, J. 1986. *Finite and Infinite Games.* Ballantine Books, Random House.

Cisco. 2012. *Visual Networking Index (VNI) Forecast (2011–2016).* http://cisco.com/en/US/netsol/ns827/networking_solutions_sub_solution.html.

Dretske, F. 1981. *Knowledge and the Flow of Information.* The MIT Press.

Gleick, J. 2011. *The Information: A History, a Theory, a Flood*. Random House.

Goldin, D., S. Smolka and P. Wegner. 2010. *Interactive Computation: The New Paradigm*. Springer.

Moore, G. 1965. Cramming more components onto integrated circuits. *Electronics* 38:8 (April), 4–6.

Rocchi, P. 2012. *The Logic of Digital and Analog Machines*. Nova Publishers.

Shannon, Claude. 1948. The mathematical theory of communication. *Bell Systems Technical Journal* 27:379–423, 623–656.

Turing, Alan M. 1937. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* 42:230–265.

von Ahn, L., and L. Dabbish. 2004. Labeling images with a computer game. In *Proceedings of the ACM Conference on Human Factors in Computing Systems.* ACM Press, pp. 319–326.

For relevant Web links, consult this issue of *American Scientist Online:*

http://www.americanscientist.org/issues/id.99/past.aspx